FEATURE CAMBIO DE CONTRASEÑA

FEATURE

@CambiarContraseña @all  
Feature: Cambiar contraseña y validar inicio de sesión en la aplicación web  
 Como usuario de la aplicación web  
 Quiero cambiar mi contraseña y validar el inicio de sesión  
 Para asegurarme de que puedo acceder a mi cuenta de manera segura  
  
 @Escenario1 @CambiarInicioLogin @Happy  
 Scenario Outline: "<Caso de prueba>" El usuario con tipo de documento "<Tipo de Documento>" cambia la contraseña y valida el inicio de sesión  
 Given que el usuario está en la página de inicio de sesión  
 When el usuario selecciona la opción de ¿Olvidaste tu contraseña?  
 And el usuario selecciona el tipo de documento "<Tipo de Documento>"  
 And el usuario ingresa el nombre de usuario "<Usuario>"  
 And el usuario hace clic en el siguiente botón  
 And el usuario hace click en el boton "Enviar correo"  
 Then el usuario debería ver el mensaje de confirmación "Te hemos enviado un correo para generar tu contraseña al siguiente correo:"  
 And el usuario ingresa a outlook con el correo "[pruebascalidad.arqti1@sura.pe](mailto:pruebascalidad.arqti1@sura.pe)" y clave "8mbKzo0b7V"  
 And el usuario búsca el último correo con el asunto "AFP Integra - Generación de contraseña"  
 And el usuario hace click al enlace de Generación de contraseña  
 And el usuario cierra el mensaje informativo  
 And el usuario ingresa una nueva contraseña "<Nueva Contraseña>"  
 And el usuario hace clic en el siguiente  
 Then el usuario debería ver el mensaje de verificación "¡Estupendo!"  
 When el usuario abre la página de inicio de sesión para probar el cambio de contraseña  
 And el usuario selecciona el tipo de documento "<Tipo de Documento>"  
 And el usuario ingresa el nombre de usuario "<Usuario>"  
 And el usuario ingresa la contraseña "<Nueva Contraseña>"  
 And el usuario hace clic en el botón de inicio de sesión  
 Then el usuario valida el mensaje de bienvenida  
  
 Examples:  
 | Caso de prueba | Tipo de Documento | Usuario | Nueva Contraseña |  
 | CP01 | DNI | 45661984 | Cercania2024 |  
 | CP02 | CE | 001108718 | Cercania2024 |  
*# | CP03 | PAS | D588394 | Cercania2024 |*  
*# | CP04 | PTP | 85808405 | Cercania2024 |*  
*# | CP05 | CRE | 09783628 | Cercania2024 |*  
*# | CP06 | CIE | 0962710232 | Cercania2024 |*  
*# | CP07 | CSR | 742402050 | Cercania2024 |*  
*# | CP08 | CPP | 1234567890 | Cercania2024 |*  
*# | CP09 | CIA | 72777884 | Cercania2024 |*  
*# | CP10 | CFA | 31974330 | Cercania2024 |*  
*#repite en login*  
  
  
  
  
  
  
  
  
  
  
@Escenario2 @NumeroIncompletoCambioContraseña @UnHappy  
 Scenario Outline: "<Caso de prueba>" El usuario con tipo de documento "<Tipo de Documento>" no puede cambiar la contraseña por usuario incompleto  
 Given que el usuario está en la página de inicio de sesión  
 When el usuario selecciona la opción de ¿Olvidaste tu contraseña?  
 And el usuario selecciona el tipo de documento "<Tipo de Documento>"  
 And el usuario ingresa el nombre de usuario "<usuario>"  
 Then el usuario debería ver el mensaje de campo user incompleto "No es un campo valido"  
  
 Examples:  
 | Caso de prueba | Tipo de Documento | usuario |  
 | CP03 | DNI | 20 |  
 | CP04 | CE | 19 |  
  
  
  
  
  
  
  
  
*#repite en login*  
  
  
  
  
  
  
@Escenario3 @UsuarioInvalidoCambiar @UnHappy  
 Scenario Outline: "<Caso de prueba>" El usuario no puede cambiar la contraseña porque el usuario es inválido  
 Given que el usuario está en la página de inicio de sesión  
 When el usuario selecciona la opción de ¿Olvidaste tu contraseña?  
 And el usuario selecciona el tipo de documento "<tipoDocumento>"  
 And el usuario ingresa el nombre de usuario "<usuario>"  
 And el usuario hace clic en el siguiente botón  
 Then el usuario debería ver el mensaje de campo user inválido "Puede que no hayas ingresado correctamente tus datos o que no seas cliente de AFP Integra."  
  
 Examples:  
 | Caso de prueba | tipoDocumento | usuario |  
 | CP05 | CE | 001108710 |  
  
  
  
  
  
  
  
  
  
  
  
 @Escenario4 @CorreoNoRegistrado @UnHappy  
 Scenario Outline: "<Caso de prueba>" El usuario no puede cambiar la contraseña porque el correo no está registrado  
 Given que el usuario está en la página de inicio de sesión  
 When el usuario selecciona la opción de ¿Olvidaste tu contraseña?  
 And el usuario selecciona el tipo de documento "<tipoDocumento>"  
 And el usuario ingresa el nombre de usuario "<usuario>"  
 And el usuario hace clic en el siguiente botón  
 Then el usuario deberia ver el mensaje de actualizar tus datos  
  
 Examples:  
 | Caso de prueba | tipoDocumento | usuario |  
 | CP06 | DNI | 01700842 |  
  
  
*#falta el escenario 5 implemetar*  
@Escenario5 @BloqueoUsuarioTercer @UnHappy  
 Scenario Outline: "<Caso de prueba>" El usuario se bloquea al 3er intento por ingresar usuario incorrecto y debe cambiar contraseña  
 Given que el usuario está en la página de inicio de sesión  
 And el usuario selecciona el tipo de documento "<Tipo de Documento>"  
 And el usuario ingresa el nombre de usuario "<Usuario>"  
 And el usuario ingresa la contraseña "abchg5g55"  
 And el usuario hace clic en el botón de inicio de sesión  
 And el usuario hace clic en el botón de inicio de sesión  
 And el usuario hace clic en el botón de inicio de sesión  
 Then el usuario debería ver el mensaje de Superaste el número de intentos  
 And el usuario hace clic en el botón de inicio de sesión  
 Then el usuario visualiza el mensaje de bloqueo temporal  
 And el usuario hace clic en el botón Cambiar contraseña  
 And el usuario hace click en el boton "Enviar correo"  
 Then el usuario debería ver el mensaje de confirmación "Te hemos enviado un correo para generar tu contraseña al siguiente correo:"  
 And el usuario ingresa a outlook con el correo "[pruebascalidad.arqti1@sura.pe](mailto:pruebascalidad.arqti1@sura.pe)" y clave "8mbKzo0b7V"  
 And el usuario búsca el último correo con el asunto "AFP Integra - Generación de contraseña"  
 And el usuario hace click al enlace de Generación de contraseña  
 And el usuario cierra el mensaje informativo  
 And el usuario ingresa una nueva contraseña "<Nueva Contraseña>"  
 And el usuario hace clic en el siguiente  
 Then el usuario debería ver el mensaje de verificación "¡Estupendo!"  
 When el usuario selecciona el tipo de documento "<Tipo de Documento>"  
 And el usuario ingresa el nombre de usuario "<Usuario>"  
 And el usuario ingresa la contraseña "<Nueva Contraseña>"  
 And el usuario hace clic en el botón de inicio de sesión  
 Then el usuario valida el mensaje de bienvenida  
  
 Examples:  
 | Caso de prueba | Tipo de Documento | Usuario | Nueva Contraseña |  
 | CP14 | DNI | 21535926 | Cercania2024 |  
 *# | CP13 | DNI | 06197878 | Cercania2024 |*  
 *# | CP13 | DNI | 04413667 | Cercania2024 |*  
 *# | CP13 | DNI | 72159283 | Cercania2024 |*  
  
 *#Superaste el número de intentos. Para acceder nuevamente haz clic en "Olvidé mi contraseña"32956445.*  
  
  
 *#escenario ejecución exitosa - Irvin*  
  
  
  
@Escenario5 @CambioAlternativo @Happy  
 Scenario Outline: "<Caso de prueba>" El usuario con tipo de documento "<Tipo de Documento>" cambia la contraseña desde la pagina principal AFP Integra  
 Given que el usuario está en la página principal  
 And el usuario acepta el cookis  
 When el usuario hace clic en la opción Crear o recuperar contraseña  
 And el usuario hace clic en el botón Cambiar contraseña desde pagina principal  
 And el usuario selecciona el tipo de documento "<Tipo de Documento>"  
 And el usuario ingresa el nombre de usuario "<Usuario>"  
 And el usuario hace clic en el siguiente botón  
 And el usuario hace click en el boton "Enviar correo"  
 Then el usuario debería ver el mensaje de confirmación "Te hemos enviado un correo para generar tu contraseña al siguiente correo:"  
 And el usuario ingresa a outlook con el correo "[pruebascalidad.arqti1@sura.pe](mailto:pruebascalidad.arqti1@sura.pe)" y clave "8mbKzo0b7V"  
 And el usuario búsca el último correo con el asunto "AFP Integra - Generación de contraseña"  
 And el usuario hace click al enlace de Generación de contraseña  
 And el usuario cierra el mensaje informativo  
 And el usuario ingresa una nueva contraseña "<Nueva Contraseña>"  
 And el usuario hace clic en el siguiente  
 Then el usuario debería ver el mensaje de verificación "¡Estupendo!"  
 When el usuario abre la página de inicio de sesión para probar el cambio de contraseña  
 And el usuario selecciona el tipo de documento "<Tipo de Documento>"  
 And el usuario ingresa el nombre de usuario "<Usuario>"  
 And el usuario ingresa la contraseña "<Nueva Contraseña>"  
 And el usuario hace clic en el botón de inicio de sesión  
 Then el usuario valida el mensaje de bienvenida  
  
 Examples:  
 | Caso de prueba | Tipo de Documento | Usuario | Nueva Contraseña |  
 | CP07 | DNI | 21535926 | Cercania2024 |  
 *# | CP01 | DNI | 00797231 | Cercania2024 |*  
 *# | CP02 | CE | 001108718 | Cercania2024 |*  
*# | CP03 | PAS | D588394 | Cercania2024 |*  
*# | CP04 | PTP | 85808405 | Cercania2024 |*  
 *# | CP05 | CRE | 09783628 | Cercania2024 |*  
 *# | CP06 | CIE | 0962710232 | Cercania2024 |*  
*# | CP07 | CSR | 742402050 | Cercania2024 |*  
*# | CP08 | CPP | 1234567890 | Cercania2024 |*  
 *# | CP09 | CIA | 72777884 | Cercania2024 |*  
 *# | CP10 | CFA | 31974330 | Cercania2024 |*

UI

package ui;  
  
import net.serenitybdd.core.annotations.findby.By;  
import net.serenitybdd.screenplay.targets.Target;  
  
public class ResetPasswordPage {  
  
 public static final Target *BTN\_RECOVER\_PASSWORD* = Target.*the*("Hacer el click en el Botón ¿Olvidé mi Contraseña?")  
 .located(By.*xpath*("//span[text()='Olvidé mi contraseña']"));  
 public static final Target *MESSAGE\_EMAIL\_VERIFICATION* = Target.*the*("Mensaje de Verificación del Correo de Electrónico")  
 .located(By.*xpath*("//div[@class='ContainerWithPersonsComponents\_c\_container\_with\_persons\_\_gGqAd i-text-center i-text-white']//div[@class='i-spacing-4x i-body i-text-white']"));  
 public static final Target *BTN\_SEARCH* = Target.*the*("Hacer click en el boton buscar")  
 .located(By.*xpath*("//\*[@title = 'Buscar']"));  
 public static final Target *SELECT\_FIRST\_EMAIL* = Target.*the*("Seleciona el correo mas reciente")  
 .locatedBy("(//div[@title='Todos los resultados' or @title='Resultados principales']/ancestor::div[@data-animatable='true']//following-sibling::div[@id])[{0}]");  
 public static final Target *INP\_NEW\_PASSWORD* = Target.*the*("Ingresa la nueva Contraseña")  
 .located(By.*id*("inpNewPassword"));  
 public static final Target *MESSAGE\_HAPPY\_VERIFICATION\_EMAIL* = Target.*the*("Mensaje de verificación del cambio de contraseña exitoso")  
 .located(By.*xpath*("//span[@class = 'i-body-medium i-text-blue']"));  
 public static final Target *BTN\_UNDERSTOOD* = Target.*the*("Mensaje de verificación del cambio de contraseña exitoso")  
 .located(By.*xpath*("//\*[@class = 'Button\_c\_button\_\_I\_x2\_ Button\_c\_button\_\_SizeMedium\_\_lNORL Button\_c\_button\_\_container\_theme\_\_uEpvR ']"));  
 public static final Target *BTN\_SHOW\_CONTENT* = Target.*the*("Boton de mostrar contenido")  
 .locatedBy("//button[text()='Mostrar contenido bloqueado']");  
 public static final Target *IMG\_LINK\_URL* = Target.*the*("Click en la imagen de ingresa aqui")  
 .located(By.*xpath*("(//\*[@data-imagetype])[2]"));  
 public static final Target *OK\_BUTTON* = Target.*the*("Click al botón Entendido")  
 .locatedBy("((//div[text()='Entendido'])/..)[3]");  
 public static final Target *CONTINUE\_BUTTON* = Target.*the*("Click al botón Entendido")  
 .locatedBy("(//div[text()='Continuar']/..)");  
  
 public static final Target *DOCUMENT\_TYPE\_DROPDOWN* = Target.*the*("Desplegable tipo de documento")  
 .located(By.*id*("sltDocumentType"));  
 public static final Target *CLICK\_OPTION\_TYPE\_DROPDOWN* = Target.*the*("Clic en la opción {0}")  
 .located(By.*xpath*("//div[@id='dropdown\_options\_sltDocumentType']//li[@data-keyprop='{0}']"));  
  
 public static final Target *MENSAJE\_SUPERASTESINTENTOS* = Target.*the*("Mensaje por usuario incorrecto")  
 .locatedBy("//p[contains(text(),'Superaste el número de intentos.')]");  
  
 public static final Target *MENSAJE\_BLOQUEO* = Target.*the*("Mensaje por usuario incorrecto")  
 .locatedBy("//div[contains(text(),'Por tu seguridad, hemos bloqueado temporalmente el ingreso a tu cuenta.')]");  
  
 public static final Target *BUTTON\_CAMBIARCONTRASEÑA* = Target.*the*("clic en botón cambiar contraseña")  
 .locatedBy("//div[text()='Cambiar contraseña']/parent::button");  
  
 public static final Target *LABEL\_GREAT* = Target.*the*("great message")  
 .locatedBy("//span[text()='¡Estupendo!']");  
  
 public static final Target *MESSAGE\_UPDATE\_DATA* = Target.*the*("validar mensaje actualizar datos")  
 .locatedBy("//h5[contains(text(),'Para actualizar tus datos validaremos tu identidad')]");  
  
  
}

STEP DEFINITIONS

package com.sura.web.stepdefinitions;  
  
import io.cucumber.java.en.And;  
import io.cucumber.java.en.Then;  
import io.cucumber.java.en.When;  
import net.serenitybdd.model.environment.EnvironmentSpecificConfiguration;  
import net.thucydides.model.environment.SystemEnvironmentVariables;  
import questions.ElementVisible;  
import questions.ResetPasswordQuestions;  
import tasks.common.\*;  
import tasks.resetPassword.\*;  
import ui.\*;  
  
import static org.hamcrest.Matchers.*is*;  
import static net.serenitybdd.screenplay.GivenWhenThen.*seeThat*;  
import static net.serenitybdd.screenplay.actors.OnStage.*theActorInTheSpotlight*;  
import static org.hamcrest.Matchers.*containsString*;  
  
public class ResetPasswordStepDefinitions {  
  
 @When("el usuario selecciona la opción de ¿Olvidaste tu contraseña?")  
 public void elUsuarioSeleccionaLaOpciónDe() {  
 // theActorInTheSpotlight().attemptsTo(new ButtonResetPassword());  
  
 *theActorInTheSpotlight*().attemptsTo(  
 WaitFor.*seconds*(5),  
 ClickElement.*on*(ResetPasswordPage.*BTN\_RECOVER\_PASSWORD*,true)  
  
 );  
  
 }  
  
 @And("el usuario envía el mensaje de verificación al correo electrónico")  
 public void elUsuarioHaceClicEnElBotónDeContinuar() {  
 *theActorInTheSpotlight*().attemptsTo(new ButtonNext());  
 }  
  
 @And("el usuario hace click en el boton {string}")  
 public void elUsuarioHaceClickEnElBoton(String text) {  
 *theActorInTheSpotlight*().attemptsTo(ButtonSendEmail.*withData*(text));  
 }  
  
 @Then("el usuario debería ver el mensaje de confirmación {string}")  
 public void elUsuarioDeberíaVerElMensajeDeConfirmación(String message) {  
 *theActorInTheSpotlight*().should(*seeThat*(ResetPasswordQuestions.*ValidateMessageConfirmation*(), *containsString*(message.trim())));  
 }  
  
 @And("el usuario hace clic en el siguiente botón")  
 public void elUsuarioHaceClicEnElSiguienteBotón() {  
 *theActorInTheSpotlight*().attemptsTo(new ButtonNext());  
 }  
  
 @And("el usuario ingresa una nueva contraseña {string}")  
 public void elUsuarioIngresaUnaNuevaContraseña(String newPassword) {  
 *theActorInTheSpotlight*().attemptsTo(InputNewPassword.*withData*(newPassword));  
 }  
  
 @Then("el usuario debería ver el mensaje de verificación {string}")  
 public void elUsuarioDeberíaVerElMensajeDeVerificación(String messageVerificationEmail) {  
  
 *theActorInTheSpotlight*().should(*seeThat*(ElementVisible.*isVisible*(  
 ResetPasswordPage.*LABEL\_GREAT*), *is*(true)));  
  
 // Get the message  
 // String message = ResetPasswordQuestions.ValidateMessageConfirmationEmail().answeredBy(theActorInTheSpotlight());  
  
 // Print the message to the console  
 // System.out.println("Confirmation Email Message: " + message);  
  
 // Store the message in a variable  
 // String confirmationMessage = message;  
 // theActorInTheSpotlight().should(seeThat(ResetPasswordQuestions.ValidateMessageConfirmationEmail(), containsString(messageVerificationEmail)));  
 // theActorInTheSpotlight().attemptsTo(new ButtonUnderStood());  
 }  
  
 @And("el usuario hace clic en el siguiente")  
 public void elUsuarioHaceClicEnElSiguiente() {  
 *theActorInTheSpotlight*().attemptsTo(new ButtonContinue());  
 }  
  
 @Then("el usuario debería ver el mensaje de Superaste el número de intentos")  
 public void elUsuarioDeberíaVerElMensajeDeSuperasteElNúmeroDeIntentos() {  
 *theActorInTheSpotlight*().should(*seeThat*(ElementVisible.*isVisible*(  
 ResetPasswordPage.*MENSAJE\_SUPERASTESINTENTOS*), *is*(true)));  
 }  
  
 @Then("el usuario visualiza el mensaje de bloqueo temporal")  
 public void elUsuarioVisualizaElMensajeDeBloqueoTemporal() {  
 *theActorInTheSpotlight*().should(*seeThat*(ElementVisible.*isVisible*(  
 ResetPasswordPage.*MENSAJE\_BLOQUEO*), *is*(true)));  
 }  
  
 @And("el usuario hace clic en el botón Cambiar contraseña")  
 public void elUsuarioHaceClicEnElBotónCambiarContraseña() {  
 *theActorInTheSpotlight*().attemptsTo(ClickElement.*on*(ResetPasswordPage.*BUTTON\_CAMBIARCONTRASEÑA*,true));  
 }  
  
  
  
  
  
  
  
 @And("el usuario cierra el mensaje informativo")  
 public void elUsuarioCierraElMensajeInformativo() {  
 *theActorInTheSpotlight*().attemptsTo(ClickElement.*on*(ResetPasswordPage.*OK\_BUTTON*,false));  
 }  
  
 @When("el usuario abre la página de inicio de sesión para probar el cambio de contraseña")  
 public void elUsuarioAbreLaPáginaDeInicioDeSesiónParaProbarElCambioDeContraseña() {  
  
 String loginurl = EnvironmentSpecificConfiguration.*from*(SystemEnvironmentVariables.*createEnvironmentVariables*())  
 .getProperty("environments.default.login.url");  
  
 *theActorInTheSpotlight*().attemptsTo(  
 OpenNewTab.*now*(),  
 NavigateToUrl.*to*(loginurl)  
 );  
 }  
  
 @When("el usuario hace clic en la opción Crear o recuperar contraseña")  
 public void elUsuarioHaceClicEnLaOpciónCrearORecuperarContraseña() {  
 // theActorInTheSpotlight().attemptsTo(WaitFor.seconds(3));  
 *theActorInTheSpotlight*().attemptsTo(ClickElementJs.*on*(HomePage.*BUTTON\_RECUPERAR\_CONTRASENA*));  
  
 }  
  
 @And("el usuario hace clic en el botón Cambiar contraseña desde pagina principal")  
 public void elUsuarioHaceClicEnElBotónCambiarContraseñaDesdePaginaPrincipal() {  
 // theActorInTheSpotlight().attemptsTo(WaitFor.seconds(3));  
 *theActorInTheSpotlight*().attemptsTo(ClickElementJs.*on*(HomePage.*BUTTON\_CAMBIAR\_CONTRASENA*));  
 }  
  
 @Then("el usuario deberia ver el mensaje de actualizar tus datos")  
 public void elUsuarioDeberiaVerElMensajeDeActualizarTusDatos() {  
 *theActorInTheSpotlight*().should(*seeThat*(ElementVisible.*isVisible*(  
 ResetPasswordPage.*MESSAGE\_UPDATE\_DATA*), *is*(true)));  
 }  
}

PASOS DE FLUJO GENERAL

@Escenario5 @CambioAlternativo @Happy

PASO 1

@Given("que el usuario está en la página principal")  
public void queElUsuarioEstáEnLaPáginaPrincipal() {  
 *theActorCalled*("user").wasAbleTo(PrincipalUrl.*CambioAFP*());  
}

package tasks.login;  
  
import net.serenitybdd.annotations.Step;  
import net.serenitybdd.model.environment.ConfiguredEnvironment;  
import net.serenitybdd.model.environment.EnvironmentSpecificConfiguration;  
import net.serenitybdd.screenplay.Actor;  
import net.serenitybdd.screenplay.Task;  
import net.serenitybdd.screenplay.actions.Open;  
import net.thucydides.model.util.EnvironmentVariables;  
  
import static net.serenitybdd.screenplay.Tasks.*instrumented*;  
  
public class PrincipalUrl implements Task {  
  
 private final String PrincipalUrl;  
 private final EnvironmentVariables environmentVariables;  
  
 public PrincipalUrl(String PrincipalUrl) {  
 this.PrincipalUrl = PrincipalUrl;  
 this.environmentVariables = ConfiguredEnvironment.*getEnvironmentVariables*();  
 }  
  
  
 public static Task CambioAFP() {  
 String PrincipalUrl = "principal.url";  
 return *instrumented*(PrincipalUrl.class, PrincipalUrl);  
 }  
  
 @Override  
 @Step("{0} Inicia la página #url")  
 public <T extends Actor> void performAs(T actor) {  
  
 String targetUrl = EnvironmentSpecificConfiguration.*from*(environmentVariables).getProperty(PrincipalUrl);  
  
 if (targetUrl == null) {  
 throw new IllegalArgumentException("URL not found in environment variables: " + PrincipalUrl);  
 } else {  
 actor.attemptsTo(Open.*url*(targetUrl));  
 }  
 }  
}

PASO 2

@And("el usuario acepta el cookis")  
public void elUsuarioAceptaElCookis() {  
 *theActorInTheSpotlight*().attemptsTo(WaitFor.*seconds*(5));  
 *theActorInTheSpotlight*().attemptsTo(Check.*whether*(CookiesPage.*COOKIES\_ACCEPT*.resolveFor(  
 *theActorInTheSpotlight*()).isVisible()).andIfSo(ClickElement.*on*(CookiesPage.*COOKIES\_ACCEPT*, true)));  
}

public static final Target *COOKIES\_ACCEPT* = Target.*the*("Accept Cookies")  
 .located(By.*id*("btn\_accept\_all\_cookies"));

PASO 3

@When("el usuario hace clic en la opción Crear o recuperar contraseña")  
public void elUsuarioHaceClicEnLaOpciónCrearORecuperarContraseña() {  
 // theActorInTheSpotlight().attemptsTo(WaitFor.seconds(3));  
 *theActorInTheSpotlight*().attemptsTo(ClickElementJs.*on*(HomePage.*BUTTON\_RECUPERAR\_CONTRASENA*));  
  
}

public static final Target *BUTTON\_RECUPERAR\_CONTRASENA* = Target.*the*("click en Botón recuperar contraseña")  
 .locatedBy("(//div[text()='Crear o recuperar tu contraseña'])[1]");

PASO 4

@And("el usuario hace clic en el botón Cambiar contraseña desde pagina principal")  
public void elUsuarioHaceClicEnElBotónCambiarContraseñaDesdePaginaPrincipal() {  
 // theActorInTheSpotlight().attemptsTo(WaitFor.seconds(3));  
 *theActorInTheSpotlight*().attemptsTo(ClickElementJs.*on*(HomePage.*BUTTON\_CAMBIAR\_CONTRASENA*));  
}

public static final Target *BUTTON\_CAMBIAR\_CONTRASENA* = Target.*the*("click en cambiar contraseña")  
 .locatedBy("//div[contains(text(),'Cambiar contraseña')]/parent::a");

PASO 5

@And("el usuario selecciona el tipo de documento {string}")  
public void elUsuarioSeleccionaElTipoDeDocumento(String typeDocument) {  
 *theActorInTheSpotlight*().attemptsTo(  
 WaitFor.*seconds*(5),  
 ClickElement.*on*(LoginPage.*DOCUMENT\_TYPE\_DROPDOWN*,false),  
 WaitFor.*seconds*(5),  
 ClickElement.*on*(LoginPage.*CLICK\_OPTION\_TYPE\_DROPDOWN*,false, typeDocument)  
 );  
}

public static final Target *DOCUMENT\_TYPE\_DROPDOWN* = Target.*the*("Mensaje por usuario incorrecto")  
 .locatedBy("//input[@id='sltDocumentType']");

public static final Target *CLICK\_OPTION\_TYPE\_DROPDOWN* = Target.*the*("Click drop down option")  
 .locatedBy("//li[@data-keyprop='{0}']");

PASO6

@And("el usuario ingresa el nombre de usuario {string}")  
public void elUsuarioIngresaElNombreDeUsuario(String user) {  
 *theActorInTheSpotlight*().attemptsTo(InputUser.*withData*(user));  
}

package tasks.login;  
  
import net.serenitybdd.annotations.Step;  
import net.serenitybdd.screenplay.Actor;  
import net.serenitybdd.screenplay.Performable;  
import net.serenitybdd.screenplay.Task;  
import net.serenitybdd.screenplay.Tasks;  
import net.serenitybdd.screenplay.actions.Enter;  
import net.serenitybdd.screenplay.matchers.WebElementStateMatchers;  
import net.serenitybdd.screenplay.waits.WaitUntil;  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
import ui.CookiesPage;  
  
public class InputUser implements Task {  
  
 private static final Logger *LOGGER* = LoggerFactory.*getLogger*(InputUser.class);  
 private final String enterUser;  
  
 public InputUser(String enterUser) {  
 this.enterUser = enterUser;  
 }  
  
 public static Performable withData(String enterUser) {  
 return Tasks.*instrumented*(InputUser.class, enterUser);  
 }  
  
 @Override  
 @Step("{0} ingresa usuario")  
 public <T extends Actor> void performAs(T actor) {  
 long endTime = System.*currentTimeMillis*() + 10000; // 10 segundos  
 boolean entered = false;  
  
 while (System.*currentTimeMillis*() < endTime && !entered) {  
 try {  
  
 if (CookiesPage.*USERNAME\_FIELD*.resolveFor(actor).isVisible()) {  
 actor.attemptsTo(  
 WaitUntil.*the*(CookiesPage.*USERNAME\_FIELD*, WebElementStateMatchers.*isClickable*()).forNoMoreThan(15).seconds(),  
 Enter.*theValue*(enterUser).into(CookiesPage.*USERNAME\_FIELD*)  
 );  
 entered = true;  
 } else {  
 *LOGGER*.info("El campo de entrada de usuario no está visible inicialmente.");  
 }  
 } catch (Exception e) {  
 *LOGGER*.warn("Intento fallido de ingresar el usuario. Reintentando...", e);  
 try {  
 Thread.*sleep*(500); // Espera medio segundo antes de reintentar  
 } catch (InterruptedException ie) {  
 Thread.*currentThread*().interrupt();  
 *LOGGER*.error("El hilo fue interrumpido", ie);  
 }  
 }  
 }  
  
 if (!entered) {  
 *LOGGER*.error("No se pudo ingresar el usuario después de 10 segundos.");  
 }  
 }  
}

public static final Target *USERNAME\_FIELD* = Target.*the*("Enter Username")  
 .located(org.openqa.selenium.By.*id* ("inpDocumentNumber"));

PASO7

@And("el usuario hace clic en el siguiente botón")  
public void elUsuarioHaceClicEnElSiguienteBotón() {  
 *theActorInTheSpotlight*().attemptsTo(new ButtonNext());  
}

package tasks.resetPassword;  
  
import net.serenitybdd.screenplay.Actor;  
import net.serenitybdd.screenplay.Task;  
import net.serenitybdd.screenplay.actions.Click;  
import net.serenitybdd.screenplay.matchers.WebElementStateMatchers;  
import net.serenitybdd.screenplay.waits.WaitUntil;  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
import ui.LoginOutlookPage;  
import ui.ResetPasswordPage;  
  
public class ButtonNext implements Task {  
  
 private static final Logger *LOGGER* = LoggerFactory.*getLogger*(ButtonNext.class);  
  
 @Override  
 public <T extends Actor> void performAs(T actor) {  
 long endTime = System.*currentTimeMillis*() + 15000; // 10 segundos  
 boolean clicked = false;  
  
 while (System.*currentTimeMillis*() < endTime && !clicked) {  
 try {  
 Thread.*sleep*(1000); // Espera medio segundo antes de reintentar  
 if (LoginOutlookPage.*NEXT\_BUTTON*.resolveFor(actor).isVisible()) {  
 actor.attemptsTo(  
 WaitUntil.*the*(LoginOutlookPage.*NEXT\_BUTTON*, WebElementStateMatchers.*isClickable*()).forNoMoreThan(15).seconds(),  
 Click.*on*(LoginOutlookPage.*NEXT\_BUTTON*)  
 );  
 clicked = true;  
 } else {  
 *LOGGER*.info("El botón Next no está visible inicialmente.");  
 }  
 } catch (Exception e) {  
 *LOGGER*.warn("Intento fallido de hacer clic en el botón Next. Reintentando...", e);  
 try {  
 Thread.*sleep*(500); // Espera medio segundo antes de reintentar  
 } catch (InterruptedException ie) {  
 Thread.*currentThread*().interrupt();  
 *LOGGER*.error("El hilo fue interrumpido", ie);  
 }  
 }  
 }  
  
 if (!clicked) {  
 *LOGGER*.error("No se pudo hacer clic en el botón Next después de 10 segundos.");  
 }  
 }  
}

public static final Target *NEXT\_BUTTON* = Target.*the*("next button")  
 .located(By.*xpath*("//\*[@type='submit']"));

PASO8

@And("el usuario hace click en el boton {string}")  
public void elUsuarioHaceClickEnElBoton(String text) {  
 *theActorInTheSpotlight*().attemptsTo(ButtonSendEmail.*withData*(text));  
}

package tasks.resetPassword;  
  
import net.serenitybdd.screenplay.Actor;  
import net.serenitybdd.screenplay.Task;  
import net.serenitybdd.screenplay.actions.Click;  
import net.serenitybdd.screenplay.actions.Scroll;  
import net.serenitybdd.screenplay.matchers.WebElementStateMatchers;  
import net.serenitybdd.screenplay.waits.WaitUntil;  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
import ui.LoginOutlookPage;  
import ui.ResetPasswordPage;  
  
public class ButtonSendEmail implements Task {  
  
 private static final Logger *LOGGER* = LoggerFactory.*getLogger*(ButtonSendEmail.class);  
 private final String clickSendEMail;  
  
 public ButtonSendEmail(String clickSendEMail) {  
 this.clickSendEMail = clickSendEMail;  
 }  
  
 public static ButtonSendEmail withData(String clickSendEMail) {  
 return new ButtonSendEmail(clickSendEMail);  
 }  
  
 @Override  
 public <T extends Actor> void performAs(T actor) {  
 long endTime = System.*currentTimeMillis*() + 15000; // 10 segundos  
 boolean clicked = false;  
  
 while (System.*currentTimeMillis*() < endTime && !clicked) {  
 try {  
 Thread.*sleep*(1000); // Espera medio segundo antes de reintentar  
 if (LoginOutlookPage.*BTN\_SEND\_EMAIL*.of(clickSendEMail).resolveFor(actor).isVisible()) {  
 actor.attemptsTo(  
 WaitUntil.*the*(LoginOutlookPage.*BTN\_SEND\_EMAIL*.of(clickSendEMail), WebElementStateMatchers.*isClickable*()).forNoMoreThan(15).seconds(),  
 Scroll.*to*(LoginOutlookPage.*BTN\_SEND\_EMAIL*.of(clickSendEMail)),  
 Click.*on*(LoginOutlookPage.*BTN\_SEND\_EMAIL*.of(clickSendEMail))  
 );  
 clicked = true;  
 } else {  
 *LOGGER*.info("El botón Send Email no está visible inicialmente.");  
 }  
 } catch (Exception e) {  
 *LOGGER*.warn("Intento fallido de hacer clic en el botón Send Email. Reintentando...", e);  
 try {  
 Thread.*sleep*(500); // Espera medio segundo antes de reintentar  
 } catch (InterruptedException ie) {  
 Thread.*currentThread*().interrupt();  
 *LOGGER*.error("El hilo fue interrumpido", ie);  
 }  
 }  
 }  
  
 if (!clicked) {  
 *LOGGER*.error("No se pudo hacer clic en el botón Send Email después de 10 segundos.");  
 }  
 }  
}

public static final Target *BTN\_SEND\_EMAIL* = Target.*the*("send email button")  
 .locatedBy("//\*[contains(text(), '{0}')]/..");

PASO9

@Then("el usuario debería ver el mensaje de confirmación {string}")  
public void elUsuarioDeberíaVerElMensajeDeConfirmación(String message) {  
 *theActorInTheSpotlight*().should(*seeThat*(ResetPasswordQuestions.*ValidateMessageConfirmation*(), *containsString*(message.trim())));  
}

public static Question<String> ValidateMessageConfirmation() {  
 return actor -> *normalizeText*(ResetPasswordPage.*MESSAGE\_EMAIL\_VERIFICATION*.resolveFor(actor).waitUntilVisible().getText().trim());  
}

public static final Target *MESSAGE\_EMAIL\_VERIFICATION* = Target.*the*("Mensaje de Verificación del Correo de Electrónico")  
 .located(By.*xpath*("//div[@class='ContainerWithPersonsComponents\_c\_container\_with\_persons\_\_gGqAd i-text-center i-text-white']//div[@class='i-spacing-4x i-body i-text-white']"));

PASO10

@And("el usuario ingresa a outlook con el correo {string} y clave {string}")  
public void elUsuarioIngresaAOutlookConElCorreoYClave(String correo, String clave) {  
  
 String outlookUrl = EnvironmentSpecificConfiguration.*from*(SystemEnvironmentVariables.*createEnvironmentVariables*())  
 .getProperty("environments.default.outlook.url");  
  
 *theActorInTheSpotlight*().attemptsTo(  
 OpenNewTab.*now*(),  
 NavigateToUrl.*to*(outlookUrl),  
 ClickElement.*on*(LoginOutlookPage.*INP\_EMAIL\_USER*,false),  
 EnterText.*into*(LoginOutlookPage.*INP\_EMAIL\_USER*,correo),  
 ClickElement.*on*(LoginOutlookPage.*NEXT\_BUTTON*,false),  
 WaitFor.*seconds*(5),  
 ClickElement.*on*(LoginOutlookPage.*INP\_PASSWORD\_USER*,false),  
 EnterText.*into*(LoginOutlookPage.*INP\_PASSWORD\_USER*,clave),  
 ClickElement.*on*(LoginOutlookPage.*NEXT\_BUTTON*,false),  
 ClickElement.*on*(LoginOutlookPage.*BTN\_YES\_NOT\_OPTION*,false, "Sí"),  
 // WaitForPageToLoad.withTimeout(10),  
 ClickElement.*on*(OutlookPage.*ACCOUNT\_1*, false)  
 // WaitForPageToLoad.withTimeout(10)  
 );  
}

public static final Target *INP\_EMAIL\_USER* = Target.*the*("Ingresar el usuario del correo")  
 .located(By.*xpath*("//input[@name='loginfmt'] | //input[@type='email']"));  
public static final Target *NEXT\_BUTTON* = Target.*the*("next button")  
 .located(By.*xpath*("//\*[@type='submit']"));  
public static final Target *BTN\_SEND\_EMAIL* = Target.*the*("send email button")  
 .locatedBy("//\*[contains(text(), '{0}')]/..");  
public static final Target *INP\_PASSWORD\_USER* = Target.*the*("Enter Password Email Outlook")  
 .locatedBy("//input[@name='passwd']");  
public static final Target *BTN\_YES\_NOT\_OPTION* = Target.*the*("Click en el botón '{0}' recordar sesión")  
 .locatedBy("//input[@value='{0}']");

public static final Target *ACCOUNT\_1* = Target.*the*("Primera cuenta")  
 .locatedBy("(//div[@role='listitem']//img)[1]");

PASO11

@And("el usuario búsca el último correo con el asunto {string}")  
public void elUsuarioBúscaElÚltimoCorreoConElAsunto(String asunto) {  
 *theActorInTheSpotlight*().attemptsTo(  
 WaitFor.*seconds*(10),  
 ClickElement.*on*(OutlookPage.*TXT\_SEARCH*,false),  
 WaitFor.*seconds*(10),  
 EnterText.*into*(OutlookPage.*TXT\_SEARCH*,asunto),  
 WaitFor.*seconds*(10),  
 ClickElement.*on*(OutlookPage.*BTN\_SEARCH*,false),  
 WaitFor.*seconds*(10),  
 SelectFirstEmail.*now*()  
 );  
}

public static final Target *TXT\_SEARCH* = Target.*the*("búsqueda")  
 .locatedBy("//input[@id='topSearchInput']");

public static final Target *BTN\_SEARCH* = Target.*the*("botón search")  
 .locatedBy("//button[@title='Buscar']");

PASO13

@And("el usuario hace click al enlace de Generación de contraseña")  
public void elUsuarioHaceClickAlEnlaceDeGeneraciónDeContraseña() {  
  
 *theActorInTheSpotlight*().attemptsTo(  
 ButtonShowContent.*now*(),  
 ClickLinkChoose.*now*(),  
 WaitFor.*seconds*(5),  
 SwitchToTab.*to*(2)  
 );  
}

package tasks.resetPassword;  
  
import net.serenitybdd.annotations.Step;  
import net.serenitybdd.screenplay.Actor;  
import net.serenitybdd.screenplay.Task;  
import net.serenitybdd.screenplay.Tasks;  
import net.serenitybdd.screenplay.actions.Click;  
import net.serenitybdd.screenplay.matchers.WebElementStateMatchers;  
import net.serenitybdd.screenplay.waits.WaitUntil;  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
import ui.ResetPasswordPage;  
  
public class ButtonShowContent implements Task {  
  
 private static final Logger *LOGGER* = LoggerFactory.*getLogger*(ButtonShowContent.class);  
  
 public static ButtonShowContent now() {  
 return Tasks.*instrumented*(ButtonShowContent.class);  
 }  
  
 @Override  
 @Step("Hacer clic en el botón Show Content si existe")  
 public <T extends Actor> void performAs(T actor) {  
 long endTime = System.*currentTimeMillis*() + 15000; // 10 segundos  
 boolean clicked = false;  
  
 while (System.*currentTimeMillis*() < endTime && !clicked) {  
 try {  
 Thread.*sleep*(500); // Espera inicial de 2000 milisegundos  
 if (ResetPasswordPage.*BTN\_SHOW\_CONTENT*.resolveFor(actor).isVisible()) {  
 actor.attemptsTo(  
 WaitUntil.*the*(ResetPasswordPage.*BTN\_SHOW\_CONTENT*, WebElementStateMatchers.*isClickable*()).forNoMoreThan(15).seconds(),  
 Click.*on*(ResetPasswordPage.*BTN\_SHOW\_CONTENT*)  
 );  
 clicked = true;  
 } else {  
 *LOGGER*.info("El botón Show Content no está visible inicialmente.");  
 }  
 } catch (Exception e) {  
 *LOGGER*.warn("Intento fallido de hacer clic en el botón Show Content. Reintentando...", e);  
 try {  
 Thread.*sleep*(500); // Espera medio segundo antes de reintentar  
 } catch (InterruptedException ie) {  
 Thread.*currentThread*().interrupt();  
 *LOGGER*.error("El hilo fue interrumpido", ie);  
 }  
 }  
 }  
  
 if (!clicked) {  
 *LOGGER*.error("No se pudo hacer clic en el botón Show Content después de 15 segundos.");  
 }  
 }  
}

public static final Target *BTN\_SHOW\_CONTENT* = Target.*the*("Boton de mostrar contenido")  
 .locatedBy("//button[text()='Mostrar contenido bloqueado']");

package tasks.resetPassword;  
  
import net.serenitybdd.annotations.Step;  
import net.serenitybdd.screenplay.Actor;  
import net.serenitybdd.screenplay.Task;  
import net.serenitybdd.screenplay.Tasks;  
import net.serenitybdd.screenplay.actions.Click;  
import net.serenitybdd.screenplay.actions.Scroll;  
import net.serenitybdd.screenplay.matchers.WebElementStateMatchers;  
import net.serenitybdd.screenplay.waits.WaitUntil;  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
import ui.ResetPasswordPage;  
  
public class ClickLinkChoose implements Task {  
  
 private static final Logger *LOGGER* = LoggerFactory.*getLogger*(ClickLinkChoose.class);  
  
 public static ClickLinkChoose now() {  
 return Tasks.*instrumented*(ClickLinkChoose.class);  
 }  
  
 @Override  
 @Step("Hacer clic en el enlace IMG\_LINK\_URL si existe")  
 public <T extends Actor> void performAs(T actor) {  
 long endTime = System.*currentTimeMillis*() + 10000; // 10 segundos  
 boolean clicked = false;  
  
 while (System.*currentTimeMillis*() < endTime && !clicked) {  
 try {  
 Thread.*sleep*(500); // Espera inicial de 500 milisegundos  
 if (ResetPasswordPage.*IMG\_LINK\_URL*.resolveFor(actor).isVisible()) {  
 actor.attemptsTo(WaitUntil.*the*(ResetPasswordPage.*IMG\_LINK\_URL*, WebElementStateMatchers.*isClickable*()).forNoMoreThan(15).seconds(), Scroll.*to*(ResetPasswordPage.*IMG\_LINK\_URL*));  
 actor.attemptsTo(WaitUntil.*the*(ResetPasswordPage.*IMG\_LINK\_URL*, WebElementStateMatchers.*isClickable*()).forNoMoreThan(15).seconds(), Click.*on*(ResetPasswordPage.*IMG\_LINK\_URL*));  
 clicked = true;  
 } else {  
 *LOGGER*.info("El enlace IMG\_LINK\_URL no está visible inicialmente.");  
 }  
 } catch (Exception e) {  
 *LOGGER*.warn("Intento fallido de hacer clic en el enlace IMG\_LINK\_URL. Reintentando...", e);  
 try {  
 Thread.*sleep*(500); // Espera medio segundo antes de reintentar  
 } catch (InterruptedException ie) {  
 Thread.*currentThread*().interrupt();  
 *LOGGER*.error("El hilo fue interrumpido", ie);  
 }  
 }  
 }  
  
 if (!clicked) {  
 *LOGGER*.error("No se pudo hacer clic en el enlace IMG\_LINK\_URL después de 10 segundos.");  
 }  
 }  
}

public static final Target *IMG\_LINK\_URL* = Target.*the*("Click en la imagen de ingresa aqui")  
 .located(By.*xpath*("(//\*[@data-imagetype])[2]"));

PASO15

@And("el usuario cierra el mensaje informativo")  
public void elUsuarioCierraElMensajeInformativo() {  
 *theActorInTheSpotlight*().attemptsTo(ClickElement.*on*(ResetPasswordPage.*OK\_BUTTON*,false));  
}

public static final Target *OK\_BUTTON* = Target.*the*("Click al botón Entendido")  
 .locatedBy("((//div[text()='Entendido'])/..)[3]");

PASO16

@And("el usuario ingresa una nueva contraseña {string}")  
public void elUsuarioIngresaUnaNuevaContraseña(String newPassword) {  
 *theActorInTheSpotlight*().attemptsTo(InputNewPassword.*withData*(newPassword));  
}

package tasks.resetPassword;  
  
import net.serenitybdd.screenplay.Actor;  
import net.serenitybdd.screenplay.Task;  
import net.serenitybdd.screenplay.actions.Enter;  
import net.serenitybdd.screenplay.matchers.WebElementStateMatchers;  
import net.serenitybdd.screenplay.waits.WaitUntil;  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
import ui.ResetPasswordPage;  
  
public class InputNewPassword implements Task {  
  
 private static final Logger *LOGGER* = LoggerFactory.*getLogger*(InputNewPassword.class);  
 private final String enterNewPassword;  
  
 public InputNewPassword(String enterNewPassword) {  
 this.enterNewPassword = enterNewPassword;  
 }  
  
 public static InputNewPassword withData(String enterNewPassword) {  
 return new InputNewPassword(enterNewPassword);  
 }  
  
 @Override  
 public <T extends Actor> void performAs(T actor) {  
 long endTime = System.*currentTimeMillis*() + 10000; // 10 segundos  
 boolean entered = false;  
  
 while (System.*currentTimeMillis*() < endTime && !entered) {  
 try {  
 if (ResetPasswordPage.*INP\_NEW\_PASSWORD*.resolveFor(actor).isVisible()) {  
 actor.attemptsTo(  
 WaitUntil.*the*(ResetPasswordPage.*INP\_NEW\_PASSWORD*, WebElementStateMatchers.*isClickable*()).forNoMoreThan(15).seconds(),  
 Enter.*theValue*(enterNewPassword).into(ResetPasswordPage.*INP\_NEW\_PASSWORD*)  
 );  
 entered = true;  
 } else {  
 *LOGGER*.info("El campo de entrada de nueva contraseña no está visible inicialmente.");  
 }  
 } catch (Exception e) {  
 *LOGGER*.warn("Intento fallido de ingresar la nueva contraseña. Reintentando...", e);  
 try {  
 Thread.*sleep*(500); // Espera medio segundo antes de reintentar  
 } catch (InterruptedException ie) {  
 Thread.*currentThread*().interrupt();  
 *LOGGER*.error("El hilo fue interrumpido", ie);  
 }  
 }  
 }  
  
 if (!entered) {  
 *LOGGER*.error("No se pudo ingresar la nueva contraseña después de 10 segundos.");  
 }  
 }  
}

public static final Target *INP\_NEW\_PASSWORD* = Target.*the*("Ingresa la nueva Contraseña")  
 .located(By.*id*("inpNewPassword"));

PASO17

@And("el usuario hace clic en el siguiente")  
public void elUsuarioHaceClicEnElSiguiente() {  
 *theActorInTheSpotlight*().attemptsTo(new ButtonContinue());  
}

package tasks.resetPassword;  
  
import net.serenitybdd.screenplay.Actor;  
import net.serenitybdd.screenplay.Task;  
import net.serenitybdd.screenplay.actions.Click;  
import net.serenitybdd.screenplay.matchers.WebElementStateMatchers;  
import net.serenitybdd.screenplay.waits.WaitUntil;  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
import ui.ResetPasswordPage;  
  
public class ButtonContinue implements Task {  
  
 private static final Logger *LOGGER* = LoggerFactory.*getLogger*(ButtonContinue.class);  
  
 @Override  
 public <T extends Actor> void performAs(T actor) {  
 long endTime = System.*currentTimeMillis*() + 10000; // 10 segundos  
 boolean clicked = false;  
  
 while (System.*currentTimeMillis*() < endTime && !clicked) {  
 try {  
 Thread.*sleep*(500); // Espera medio segundo antes de inciar  
 if (ResetPasswordPage.*CONTINUE\_BUTTON*.resolveFor(actor).isVisible()) {  
 actor.attemptsTo(  
 WaitUntil.*the*(ResetPasswordPage.*CONTINUE\_BUTTON*, WebElementStateMatchers.*isClickable*()).forNoMoreThan(15).seconds(),  
 Click.*on*(ResetPasswordPage.*CONTINUE\_BUTTON*)  
 );  
 clicked = true;  
 } else {  
 *LOGGER*.info("El botón Continuar no está visible inicialmente.");  
 }  
 } catch (Exception e) {  
 *LOGGER*.warn("Intento fallido de hacer clic en el botón Continuar. Reintentando...", e);  
 try {  
 Thread.*sleep*(500); // Espera medio segundo antes de reintentar  
 } catch (InterruptedException ie) {  
 Thread.*currentThread*().interrupt();  
 *LOGGER*.error("El hilo fue interrumpido", ie);  
 }  
 }  
 }  
  
 if (!clicked) {  
 *LOGGER*.error("No se pudo hacer clic en el botón Continuar después de 10 segundos.");  
 }  
 }  
}

public static final Target *CONTINUE\_BUTTON* = Target.*the*("Click al botón Entendido")  
 .locatedBy("(//div[text()='Continuar']/..)");

PASO18

@Then("el usuario debería ver el mensaje de verificación {string}")  
public void elUsuarioDeberíaVerElMensajeDeVerificación(String messageVerificationEmail) {  
  
 *theActorInTheSpotlight*().should(*seeThat*(ElementVisible.*isVisible*(  
 ResetPasswordPage.*LABEL\_GREAT*), *is*(true)));  
  
 // Get the message  
 // String message = ResetPasswordQuestions.ValidateMessageConfirmationEmail().answeredBy(theActorInTheSpotlight());  
  
 // Print the message to the console  
// System.out.println("Confirmation Email Message: " + message);  
  
 // Store the message in a variable  
 // String confirmationMessage = message;  
 // theActorInTheSpotlight().should(seeThat(ResetPasswordQuestions.ValidateMessageConfirmationEmail(), containsString(messageVerificationEmail)));  
// theActorInTheSpotlight().attemptsTo(new ButtonUnderStood());  
}

public static final Target *LABEL\_GREAT* = Target.*the*("great message")  
 .locatedBy("//span[text()='¡Estupendo!']");

PASO19

@When("el usuario abre la página de inicio de sesión para probar el cambio de contraseña")  
public void elUsuarioAbreLaPáginaDeInicioDeSesiónParaProbarElCambioDeContraseña() {  
  
 String loginurl = EnvironmentSpecificConfiguration.*from*(SystemEnvironmentVariables.*createEnvironmentVariables*())  
 .getProperty("environments.default.login.url");  
  
 *theActorInTheSpotlight*().attemptsTo(  
 OpenNewTab.*now*(),  
 NavigateToUrl.*to*(loginurl)  
 );  
}

PASO20

@And("el usuario selecciona el tipo de documento {string}")  
public void elUsuarioSeleccionaElTipoDeDocumento(String typeDocument) {  
 *theActorInTheSpotlight*().attemptsTo(  
 WaitFor.*seconds*(5),  
 ClickElement.*on*(LoginPage.*DOCUMENT\_TYPE\_DROPDOWN*,false),  
 WaitFor.*seconds*(5),  
 ClickElement.*on*(LoginPage.*CLICK\_OPTION\_TYPE\_DROPDOWN*,false, typeDocument)  
 );  
}

public static final Target *DOCUMENT\_TYPE\_DROPDOWN* = Target.*the*("Mensaje por usuario incorrecto")  
 .locatedBy("//input[@id='sltDocumentType']");

public static final Target *CLICK\_OPTION\_TYPE\_DROPDOWN* = Target.*the*("Click drop down option")  
 .locatedBy("//li[@data-keyprop='{0}']");

PASO 21

@And("el usuario ingresa el nombre de usuario {string}")  
public void elUsuarioIngresaElNombreDeUsuario(String user) {  
 *theActorInTheSpotlight*().attemptsTo(InputUser.*withData*(user));  
}

package tasks.login;  
  
import net.serenitybdd.annotations.Step;  
import net.serenitybdd.screenplay.Actor;  
import net.serenitybdd.screenplay.Performable;  
import net.serenitybdd.screenplay.Task;  
import net.serenitybdd.screenplay.Tasks;  
import net.serenitybdd.screenplay.actions.Enter;  
import net.serenitybdd.screenplay.matchers.WebElementStateMatchers;  
import net.serenitybdd.screenplay.waits.WaitUntil;  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
import ui.CookiesPage;  
  
public class InputUser implements Task {  
  
 private static final Logger *LOGGER* = LoggerFactory.*getLogger*(InputUser.class);  
 private final String enterUser;  
  
 public InputUser(String enterUser) {  
 this.enterUser = enterUser;  
 }  
  
 public static Performable withData(String enterUser) {  
 return Tasks.*instrumented*(InputUser.class, enterUser);  
 }  
  
 @Override  
 @Step("{0} ingresa usuario")  
 public <T extends Actor> void performAs(T actor) {  
 long endTime = System.*currentTimeMillis*() + 10000; // 10 segundos  
 boolean entered = false;  
  
 while (System.*currentTimeMillis*() < endTime && !entered) {  
 try {  
  
 if (CookiesPage.*USERNAME\_FIELD*.resolveFor(actor).isVisible()) {  
 actor.attemptsTo(  
 WaitUntil.*the*(CookiesPage.*USERNAME\_FIELD*, WebElementStateMatchers.*isClickable*()).forNoMoreThan(15).seconds(),  
 Enter.*theValue*(enterUser).into(CookiesPage.*USERNAME\_FIELD*)  
 );  
 entered = true;  
 } else {  
 *LOGGER*.info("El campo de entrada de usuario no está visible inicialmente.");  
 }  
 } catch (Exception e) {  
 *LOGGER*.warn("Intento fallido de ingresar el usuario. Reintentando...", e);  
 try {  
 Thread.*sleep*(500); // Espera medio segundo antes de reintentar  
 } catch (InterruptedException ie) {  
 Thread.*currentThread*().interrupt();  
 *LOGGER*.error("El hilo fue interrumpido", ie);  
 }  
 }  
 }  
  
 if (!entered) {  
 *LOGGER*.error("No se pudo ingresar el usuario después de 10 segundos.");  
 }  
 }  
}

public static final Target *USERNAME\_FIELD* = Target.*the*("Enter Username")  
 .located(org.openqa.selenium.By.*id* ("inpDocumentNumber"));

PASO 22

@And("el usuario ingresa la contraseña {string}")  
public void elUsuarioIngresaLaContraseña(String password) {  
 *theActorInTheSpotlight*().attemptsTo(InputPassword.*withData*(password));  
}

package tasks.login;  
  
import net.serenitybdd.screenplay.Actor;  
import net.serenitybdd.screenplay.Task;  
import net.serenitybdd.screenplay.actions.Enter;  
import net.serenitybdd.screenplay.matchers.WebElementStateMatchers;  
import net.serenitybdd.screenplay.waits.WaitUntil;  
import ui.CookiesPage;  
  
public class InputPassword implements Task {  
  
 private final String enterPassword;  
  
 public InputPassword(String enterPassword) {  
 this.enterPassword = enterPassword;  
 }  
  
 public static InputPassword withData(String enterPassword) {  
 return new InputPassword(enterPassword);  
 }  
  
 @Override  
 public <T extends Actor> void performAs(T actor) {  
 actor.attemptsTo(  
 WaitUntil.*the*(CookiesPage.*PASSWORD\_FIELD*, WebElementStateMatchers.*isVisible*()).forNoMoreThan(15).seconds(),  
 Enter.*theValue*(enterPassword).into(CookiesPage.*PASSWORD\_FIELD*));  
 }  
}

public static final Target *PASSWORD\_FIELD* = Target.*the*("Enter Password")  
 .located(org.openqa.selenium.By.*id*("inpPassword"));

PASO 23

@And("el usuario hace clic en el botón de inicio de sesión")  
public void elUsuarioHaceClicEnElBotónDeInicioDeSesión() {  
 *theActorInTheSpotlight*().attemptsTo(  
 ClickElement.*on*(LoginPage.*LOGIN\_BUTTON*,false)  
 );  
}

public static final Target *LOGIN\_BUTTON* = Target.*the*("Click on the login button")  
 .located(By.*xpath*("//button[@type='submit']"));

PASO 24

@Then("el usuario valida el mensaje de bienvenida")  
public void elUsuarioValidaElMensajeDeBienvenida() {  
 *theActorInTheSpotlight*().should(  
 *seeThat*(  
 "El mensaje de bienvenida es correcto",  
 LoginQuestions.*ValidateMessage*(),  
 *anyOf*(  
 *containsString*("Bienvenido a tu Agencia Digital"),  
 *containsString*("Bienvenida a tu Agencia Digital")  
 )  
 )  
 );  
}

public static Question<String> ValidateMessage() {  
 return actor -> CookiesPage.*MESSAGE\_WELCOME\_LOGIN*.resolveFor(actor).waitUntilVisible().getText().trim();  
}

public static final Target *MESSAGE\_WELCOME\_LOGIN* = Target.*the*("Message Welcome")  
 .locatedBy("//div[@class='i-subtitle']");